

# C++ pour la robotique

C++ - Cours 2

Sébastien Rothhut





## Dans l'épisode précédent...

C++ : langage compilé, héritier du C

Organisation en fichiers .h / .hpp (headers) et .cpp

Fonction `main( )` : point d'entrée du programme



## Dans l'épisode précédent...

Types de base : numériques (**short, int, long, float, double**),  
**char, bool**

Fonctions et opérateurs d'entrée / sortie :

- `std::cin >> x;`
- `std::cout << "Hello World";`

Structures de contrôle : **if / else, while, for**



## Exercice



[https://www.onlinegdb.com/online\\_c++\\_compiler/](https://www.onlinegdb.com/online_c++_compiler/)

Écrire un programme qui demande à l'utilisateur de taper un entier N entre 0 et 20 bornes incluses et qui affiche N+17.

Si on tape une valeur erronée, il faut afficher "erreur" et demander de saisir à nouveau l'entier.



## Exercice

Écrire un programme qui **demande à l'utilisateur de taper un entier N entre 0 et 20 bornes incluses** et qui **affiche N+17**.

Si on tape une valeur erronée, il faut **afficher "erreur"** et **demander de saisir à nouveau l'entier**.



# Exercice

```
#include<iostream>
using namespace std;

int main() {
    int nn;

    while (1) {
        cout<<"Entrer un entier entre 0 et 20 inclusif :";
        cin >> nn;
        if ((nn>=0) && (nn<=20)) break;
        cout << " erreur\n";
    }
    cout << "\n";

    cout << nn << " + 17 = " << nn+17;
    return 0;
}
```



## Exercice

Écrire un programme qui demande à l'utilisateur de saisir un entier N et qui affiche la figure suivante.

N=1

\*

N=2

\*\*

\*

N=3

\*\*\*

\*\*

\*

...



# Exercice

```
#include <iostream>
```

```
using namespace std;
```

```
int main() {  
    int N=0;  
    cout<<"Saisissez une valeur pour N: ";  
    cin >> N;  
    cout<<endl;  
    cout<<"N="<<N<<endl;  
    for(int i=0; i<N; i++) {  
        for(int j=i; j<N; j++) {  
            cout<<"*";  
        }  
        cout<<endl;  
    }  
    return 0;  
}
```





## Exercice

Écrire un programme qui demande à l'utilisateur de saisir un entier N et qui affiche la figure suivante.

N == 1

\*

N == 2

\*\*

\*

N == 3

\*\*\*

\*\*

\*

...



# Exercice

```
#include<iostream>
using namespace std;

int main() {
    int i,j,N;

    cout<<"Tapez la valeur de N : ";
    cin >> N;

    for(i=0;i<N;i++) {
        for(j=0; j<i; j++) cout<<" ";
        for(; j<N; j++) cout<<"*";
        cout<<endl;
    }
    return 0;
}
```



# Fonctions





# Fonctions

Déclaration :

```
int function(char c, bool b);
```

- nom de la fonction
  - paramètres
  - type de retour (peut être **void**)
- } **signature** de la fonction



# Fonctions

Déclaration :

```
int function(char c, bool b);
```

→ **prototype** de la fonction, qu'on retrouve dans le fichier .h  
/ .hpp



# Fonctions

Définition : bloc d'instructions qui vont être effectivement exécutées → **corps** de la fonction


```
/**  
 * Retourne la valeur maximum des deux entiers en argument  
 */  
int maximum(int a, int b) {  
    if (a >= b) {  
        return a;  
    } else {  
        return b;  
    }  
}
```



# Appel de fonctions



```
int maxArray() {  
    int array[5] = {1, 20, 6, 34, 3};  
    int i;  
    for (i = 0; i < 5; i++) {  
        t = array[i];  
    }  
    return t;  
}  
  
int max(int a, int b) {  
    if (a > b) {  
        return a;  
    } else {  
        return b;  
    }  
}
```





# Appel de fonctions

```
int maximum(int a, int b);

int maxArray() {
    int array[5] = {1, 20, 6, 34, 3};
    int result = array[0];
    for (int i = 1; i < 5; ++i) {
        result = maximum(result, array[i]);
    }
    return result;
}

int maximum(int a, int b) {
    if (a >= b) {
        return a;
    } else {
        return b;
    }
}
```





## Exercice

Ecrire la méthode maximum en le moins de lignes possible

Prototype : `int maximum(int a, int b);`



# Exercise

```
int maximum(int a, int b) {  
    return (a > b ? a : b);  
}
```



# Paramètres de fonctions

Passage par **valeur** : c'est ce qu'on a vu jusqu'à présent.  
C'est un passage par **copie**

→ toute modification des paramètres reste locale à la fonction

```
int maximum(int a, int b) {  
    return (a > b ? a : b);  
}
```

```
int i = 5, j = 4;  
int m = maximum(i, j);
```



# Paramètres de fonctions

Mais si on veut modifier les valeurs des variables ?

Ecrire une fonction qui échange les valeurs des deux variables en paramètre : `void echange(int a, int b);`

```
void echange(int a, int b) {  
    int aux;  
    aux = a; a = b; b = aux;  
}  
  
int i = 3;  
int j = 5;  
echange(i, j);
```



# Paramètres de fonctions

Comment modifier les valeurs des paramètres passés à une fonction ?

Passage par **pointeur** : passage d'un pointeur sur la variable (qui contient son adresse)

```
void echange(int *a, int *b) {  
    int aux;  
    aux = *a; *a = *b; *b = aux;  
}  
  
int i = 3;  
int j = 5;  
echange(&i, &j);
```



# Paramètres de fonctions

Comment modifier les valeurs des paramètres passés à une fonction ?

Passage par **référence** : équivaut à un synonyme de la variable. On ne passe pas une copie mais bel et bien l'objet donné en argument, via sa référence

```
void echange(int &a, int &b) {  
    int aux;  
    aux = a; a = b; b = aux;  
}  
  
int i = 3;  
int j = 5;  
echange(i, j);
```



## Exercice


Écrire une fonction ayant en paramètre un entier et qui renvoie true si l'entier est premier et false sinon. Tester cette fonction.

Prototype : `bool estPremier(int n);`

**Note :** Pour tester si  $n$  ( $\geq 2$ ) est premier, il suffit de tester sa divisibilité par les entiers dont le carré est inférieur ou égal à  $n$ .

**Note 2 :** 1 n'est pas premier par définition

**Note 3 :** On supposera que l'utilisateur a entré un entier



```
#include<iostream>
using namespace std;

bool estPremier(int n) {
    if (n < 2) {
        return false; // par définition
    }
    int d=2;
    while(d*d <= n) {
        if(n%d == 0) {
            return false;
        } else {
            d++;
        }
    }
    return true;
}
```

```
int main() {
    int n;
    bool premier;
    while (true) {
        cout<<"Tapez n : ";
        cin>>n;
        if (n>0) {
            break;
        }
    }
    premier = estPremier(n);
    if(premier) {
        cout<<n<<" est premier"<<endl;
    } else {
        cout<<n<<" n'est pas premier"<<endl;
    }
    return 0;
}
```





# Questions ?

